Discovering outstanding subgroup lists for numeric targets using MDL

Manuel Proença, H.; Grünwald, P.D.; Bäck, T.H.W.; Leeuwen, M. van; Hutter, F.; Kersting, K.; ... ; Valera, I.

Discovering Outstanding Subgroup Lists for Numeric Targets Using MDL

Hugo M. Proença[1(✉)], Peter Grünwald[1,2], Thomas Bäck[1], and Matthijs van Leeuwen[1]

[1] Leiden University, Leiden, Netherlands
{h.manuel.proenca,T.H.W.Baeck,m.van.leeuwen}@liacs.leidenuniv.nl
[2] CWI, Amsterdam, Netherlands
Peter.Grunwald@cwi.nl

Abstract. The task of subgroup discovery (SD) is to find interpretable descriptions of subsets of a dataset that stand out with respect to a target attribute. To address the problem of mining large numbers of redundant subgroups, subgroup set discovery (SSD) has been proposed. State-of-the-art SSD methods have their limitations though, as they typically heavily rely on heuristics and/or user-chosen hyperparameters.

We propose a dispersion-aware problem formulation for subgroup set discovery that is based on the minimum description length (MDL) principle and subgroup lists. We argue that the best subgroup list is the one that best summarizes the data given the overall distribution of the target. We restrict our focus to a single numeric target variable and show that our formalization coincides with an existing quality measure when finding a single subgroup, but that—in addition—it allows to trade off subgroup quality with the complexity of the subgroup. We next propose SSD++, a heuristic algorithm for which we empirically demonstrate that it returns outstanding subgroup lists: non-redundant sets of compact subgroups that stand out by having strongly deviating means and small spread.

Keywords: Pattern mining · Subgroup discovery · The MDL principle

1 Introduction

Subgroup discovery [2,9] (SD) is the task of discovering subsets of the data that stand out with respect to a given target. It has a wide range of applications in many different domains [17]. For example, insurance companies could use it for fraud detection, where a found subgroup '$provider$ = HospitalX $\wedge$ $care$ = leg in cast $\rightarrow$ $average(claim)$ = \$2829.50' might indicate that a certain health care provider claims much more for certain care than others.

Since its conception subgroup discovery has been developed for various types of data and targets, e.g., nominal, numeric, and multi-label [11] targets. In this paper we limit the scope to attribute-value data with a numeric target, i.e., each data point is a row with exactly one value for each attribute and a single, numeric target label, as is also considered in the regular regression setting.

F. Hutter et al. (Eds.): ECML PKDD 2020, LNAI 12457, pp. 19–35, 2021.
https://doi.org/10.1007/978-3-030-67658-2_2

Related Work. Subgroup discovery traditionally focused on mining the top-k subgroups, based on their individual qualities. This approach has two major drawbacks: 1) its focus on quality measures that only take into account the centrality measure of the subgroup, such as the mean or median, and 2) the *pattern explosion*, i.e., typically large amounts of redundant patterns are found.

In response to the centrality problem of numeric targets, dispersion-aware measures—that allow for efficient mining of the top-k patterns—were proposed [4], but these do not address the second drawback, i.e., the pattern explosion.

To address this drawback, methods for *subgroup set discovery* (SSD) have emerged. While SD aims on ranking the quality of subgroups regardless of how they cover the data together, SSD aims at finding good quality subgroups that together describe different regions of the data with minimum overlap between those. However, most of the SSD methods focus on binary target variables [3,5,10]. For the setting with a numerical target variable, three approaches have been proposed:

1) Sequential covering: CN2-SD [10], originally introduced for nominal targets, can be directly applied to numeric targets. The idea is to iteratively find the subgroup with the highest quality, removing the data covered by that subgroup, and repeating this process until no further subgroups are found. This is virtually the same as mining a list of subgroups and therefore closest to our approach.

2) Diverse Subgroup Set Discovery (DSSD) [12]: DSSD uses a diverse beam search to find a non-redundant set of high-quality subgroups. It is based on a two-step approach that first mines a large pool of subgroups based on their individual qualities and then selects subgroups from that pool that maximize quality while penalizing for overlap. DSSD relies on tunable hyperparameters for the search and overlap penalization, which strongly influence the results.

3) Subjectively interesting Subgroup Discovery (SISD) [13]: This approach finds the subjectively most interesting subgroup with regard to the prior knowledge of the user, based on an information-theoretic framework. By successively updating the prior knowledge based on the found subgroups, it iteratively mines a diverse set of subgroups that are also dispersion-aware.

Apart from the limitations already mentioned, all three approaches lack a *global* formalization of the optimal set of subgroups for a given dataset and instead employ a sequential approach for which the stopping criteria, such as the total number of patterns to be found, need to be manually defined.

Contributions.[1] We introduce a principled approach for dispersion-aware subgroup set discovery that builds on recent work [18,22] that uses the minimum description length (MDL) principle [8,19] for pattern-based modelling. The MDL principle states that the best model is the one that compresses the data and model best and is ideally suited for model selection tasks where the goal is to find succinct and descriptive models—such as is the case in subgroup discovery.

[1] The **extended version** of this work is available on arXiv [16].

Table 1. First 4 subgroups of a subgroup list obtained by SSD++ on the *Hotel booking* dataset with target *lead days*—number of days in advance the bookings were done (this case study is discussed in Sect. 6). *Description* contains information regarding client bookings, n the number of instances covered, $\hat{\mu}$ and $\hat{\sigma}$ are the mean and standard deviation in days, and *overlap* is the percentage of the subgroup description that is covered by subgroups that come before in the list, i.e., how independently can the subgroups be interpreted. The last line represents the dataset overall probability distribution. * The n of the dataset is the total number of instances in the dataset.

s	**description** of client bookings	n	$\hat{\mu}$	$\hat{\sigma}$	overlap
1	month = 9 & customer_type = Transient-Party & meal = Half Board & country = GBR & adults ≥ 2	22	533	34	–
2	month $\in [7, 9]$ & market_segment = Groups & weekend_nights = 1 & distribution_channel = Direct	29	336	~ 0	0%
3	month = 9 & week_nights =4 & distribution_channel = Corporate	16	343	3	0%
4	week_nights = 0 & deposit_type = Refundable & repeated_guest = no & adults≥ 2	20	9	~ 0	0%
dataset overall distribution		18 550*	92	99	–

Our three main contributions are: *1)* A formalization of subgroup set discovery for numeric targets using the MDL principle. To this end we devise a model class based on probabilistic rule lists. This probabilistic approach not only enables MDL-based model selection, naturally identifying compact subgroup lists, but also takes into account the dispersion (or spread) of the target value. By mining an ordered list of subgroups rather than an unordered set, we avoid the problem of a single instance being covered by multiple subgroups. This comes at the cost of slightly reduced interpretability, as the subgroups always need to be considered in order, but note that the still often-used sequential covering approach effectively identifies subgroup lists as well. *2)* Derivations that show how our formalization relates to both an existing subgroup quality measure and Bayesian testing, and—based on these insights—a novel evaluation measure for subgroup lists. *3)* SSD++, a heuristic algorithm that finds a set of non-redundant patterns according to our MDL-based problem formulation.

Example. To illustrate how our MDL-based problem formulation naturally defines a succinct and non-redundant set of subgroups for a given dataset, without the need to define the desired diversity or number of patterns in advance, we show an example subgroup list as obtained by our approach on the *Hotel booking* dataset (see Table 1 for the details and in depth explanation in Sect. 6). Our method identifies a detailed list of booking descriptions from which we show here the first four subgroups, each consisting of a short description that clearly represent different sub-populations of the data, i.e., different types of client bookings.

2 Subgroup Discovery with Numeric Targets

Consider a dataset $D = (X, Y) = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_n, y_n)\}$. Each example $(\mathbf{x}_i, y_i)$ is composed of a numeric target value y_i and an instance of values of the explanatory variables $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{ik})$. Each instance value x_{ij} is associated to variable v_j and the total number of values in an instance is $k = |V|$ values, one for each variable v_j in V, which represents the set of all explanatory variables present in X. The domain of a variable v_j, denoted $\mathcal{X}_j$, can be one of three types: numeric, binary, or nominal (with $>$ 2values). Y is a vector of values y_i of the numeric target variable with domain $\mathcal{Y} = \mathbb{R}$.

Subgroups. A subgroup, denoted by s, consists of a *description* (also intent) that defines a *cover* (also extent), i.e., a subset of dataset D.

Subgroup Description. A description a is a Boolean function over all explanatory variables V. Formally, it is a function $a : \mathcal{X}_1 \times \cdots \mathcal{X}_{|V|} \mapsto \{false, true\}$. In our case, a description a is a conjunction of conditions on V, each specifying a specific value or interval on a variable. The domain of possible conditions depends on the type of a variable: numeric variables support *greater and less than* $\{\geq, \leq\}$; binary and categorical support *equal to* $\{=\}$. The size of a pattern a, denoted $|a|$, is the number of variables it contains. In Table 1, subgroup 1 has description of size $|a| = 5$, where two of those conditions are $\{\text{meal} = \text{Half Board}\}$ and $\{\text{adult} \geq 2\}$; on a categorical and a numerical variable, respectively.

Subgroup Cover. The cover is the bag of instances from D where the subgroup description holds true. Formally, it is defined by $D_a = \{(\mathbf{x}, y) \in D \mid a \sqsubseteq \mathbf{x}\}$, where we use $a \sqsubseteq \mathbf{x}$ to denote $a(\mathbf{x}) = true$. Further, let $|D_a|$ denote the coverage of the subgroup, i.e., the number of instances it covers.

Interpretation as Probabilistic Rule. As D_a encompasses both the explanatory variables and the target variable, the effect of a on the target variable can be interpreted as a probabilistic rule $a \mapsto \hat{f}_a(Y)$ that associates the antecedent a to its corresponding target values in Y through the empirical distribution of their values $\hat{f}_a(y)$. Note that in general $\hat{f}_a(Y)$ can be described by a statistical model and corresponding statistics $\hat{\Theta}$, e.g., a normal distribution $\mathcal{N}$ with given mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$.

Revisiting the subgroup list in Table 1, the description and corresponding statistics for the third subgroup are $a = \{\text{month} = 9 \;\&\; \text{week\_nights} = 4 \;\&\; \text{distribution\_channel} = \text{Corporate}\}$ and $\hat{\Theta}_a = \{\hat{\mu} = 343; \hat{\sigma} = 3\}$, respectively, and together represent the following rule:

$$\text{IF } a \sqsubseteq \mathbf{x} \text{ THEN lead time} \sim \mathcal{N}(\mu = 343; \sigma = 3)$$

where $\mathcal{N}(\mu; \sigma)$ is the probability density function of a normal distribution.

Quality Measures. To assess the quality (or interestingness) of a subgroup description a, a measure that scores subsets D_a needs to be chosen. The measures used vary depending on the target and task [2], but for a numeric target it usually has two components: 1) representativeness of the subgroup in the data, based

on coverage $|D_a|$; and 2) a function of the difference between a statistic of the empirical target distribution of the pattern, $\hat{f}_a(Y)$, and the overall empirical target distribution of the dataset, $\hat{f}_d(Y)$. The latter corresponds to the statistics estimated over the whole data, e.g., in Table 1 it is $\hat{\Theta}_d = \{\hat{\mu} = 92; \hat{\sigma} = 99\}$ and it is estimated over all 18 550 instances of the dataset.

The general form of a quality measure to be maximized is

$$q(a) = |D_a|^{\alpha} g(\hat{f}_a(Y), \hat{f}_d(Y)),\ \alpha \in [0, 1], \tag{1}$$

where α allows to control the trade-off between coverage and the difference of the distributions, and $g(\hat{f}_a(y), \hat{f}_d(y))$ is a function that measures how different the subgroup and dataset distributions are. The most adopted quality measure is the Weighted Relative Accuracy (WRAcc) [2], with $\alpha = 1$ and $g(\hat{f}_a(Y), \hat{f}_d(Y)) = \hat{\mu}_a - \hat{\mu}_d$ (the difference between averages of subgroup and dataset).

Subgroup Set Discovery. Subgroup set discovery [12] is the task of finding a set of high-quality, non-redundant subgroups that together describe all substantial deviations in the target distribution. That is, given a quality function Q for subgroup sets and the set of all possible subgroup sets $\mathcal{S}$, the task is to find that subgroup set $S^* = \{s_1, \ldots, s_k\}$ given by $S^* = \arg\max_{S \in \mathcal{S}} Q(S)$.

Ideally this measure should 1) *be global*, i.e., for a given dataset it should be possible to compare subgroup set qualities regardless of subgroup set size or coverage; 2) *maximize the individual qualities* of the subgroups; and 3) *minimize redundancy* of the subgroup set, i.e., the subgroups covers should overlap as little as possible while ensuring 2.

3 MDL-Based Subgroup Set Discovery

In this section we formalize the task of subgroup set discovery as a model selection problem using the Minimum Description Length (MDL) principle [8,19]. To this end we first need to define an appropriate model class $\mathcal{M}$; as we will explain next, we use *subgroup lists* as our models. The model selection problem should then be formalized using a two-part code [8], i.e.,

$$M^* = \arg\min_{M \in \mathcal{M}} L(D, M) = \arg\min_{M \in \mathcal{M}} \left[L(Y \mid X, M) + L(M) \right], \tag{2}$$

where $L(Y \mid X, M)$ is the encoded length, in bits[2], of target Y given explanatory data X and model M, and $L(M)$ is the encoded length, in bits, of the model. Intuitively, the best model M^* is that model that results in the best trade-off between how well the model compresses the target data and the complexity of that model—thus minimizing redundancy and automatically selecting the best subgroup list size. This formulation is similar to those previously used for two-view association discovery and multi-class classification [18,21]. We will first describe the details of the model class and then the required length functions.

[2] To obtain code lengths in bits, all logarithms in this paper are to the base 2.

3.1 Model Class: Subgroup Lists

Although Eq. (2) provides a *global* criterion that enables the comparison of subgroup sets of different sizes, subgroups are descriptions of *local* phenomena and we require each *individual subgroup to have high quality*.

We can accomplish this by using *subgroup lists* as models; see Eq. (3). Specifically, as we are only interested in finding subgroups for which the target deviates from the overall distribution, we assume y values to be distributed according to $\hat{f}_d$ by default (last line in Eq. (3)). For each region in the data for which the target distribution deviates from that distribution and a description exists, a subgroup specifying a different distribution $\hat{f}_a$ is added to the list.

We model the empirical distributions $\hat{f}$ by normal distributions, as those capture the two properties of interest, i.e., centre and spread, while being robust to cases where f violates the normality assumption [8]. We thus define $\hat{f}_{\hat{\mu},\hat{\sigma}}(y) = (2\pi\hat{\sigma})^{-1/2} \exp \frac{(y-\hat{\mu})^2}{2\hat{\sigma}^2}$, where $\hat{\mu}$ and $\hat{\sigma}$ are the estimated mean and standard deviation, respectively. These statistics can be easily estimated using the maximum likelihood estimator, so that a pattern a establishes a rule of the form $\text{IF } a \sqsubseteq \mathbf{x} \text{ THEN } \mathcal{N}(\hat{\mu}_i, \hat{\sigma}_i)$. Combining subgroup distributions $\hat{f}_{a,\hat{\mu}_a,\hat{\sigma}_a}$ with estimated dataset distribution $\hat{f}_{d,\hat{\mu}_d,\hat{\sigma}_d}$, this leads to a subgroup list M given by

$$\begin{aligned} \text{subgroup 1} &: \text{IF } a_1 \sqsubseteq \mathbf{x} \text{ THEN } \hat{f}_{a_1,\hat{\mu}_1,\hat{\sigma}_1}(y) \\ &\vdots \\ \text{subgroup k} &: \text{ELSE IF } a_k \sqsubseteq \mathbf{x} \text{ THEN } \hat{f}_{a_k,\hat{\mu}_k,\hat{\sigma}_k}(y) \\ \text{dataset} &: \text{ELSE } \hat{f}_{d,\hat{\mu}_d,\hat{\sigma}_d}(y) \end{aligned} \tag{3}$$

This corresponds to a probabilistic rule list with $k = |S|$ subgroups and a last (default) rule which is fixed to the overall empirical distribution $\hat{f}_{d,\hat{\mu},\hat{\sigma}}$ [18]. Fixing the distribution of this last 'rule' is crucial and differentiates a subgroup list from rule lists as used in classification and/or regression, as this enforces the discovery of a set of subgroups that individually all have target distributions that substantially deviate from the overall target distribution.

3.2 Model Encoding

The next step is to define the two length functions; we start with $L(M)$. Following the MDL principle [8], we need to ensure that 1) all models in the model class, i.e., all subgroup lists for a given dataset, can be distinguished; and 2) larger code lengths are assigned to more complex models. To accomplish the former we encode all elements of a model that can change, while for the latter we resort to two different codes: when a larger value represents a larger complexity we use the universal code for integers [8], denoted[3] $L_{\mathbb{N}}$, and when we have no prior knowledge but need to encode an element from a set we choose the uniform code.

[3] $L_{\mathbb{N}}(i) = \log k_0 + \log^* i$, where $\log^* i = \log i + \log\log i + \ldots$ and $k_0 \approx 2.865064$.

Specifically, the encoded length of a model M over variables V is given by

$$L(M) = L_{\mathbb{N}}(|S|) + \sum_{a_i \in S} \left[L_{\mathbb{N}}(|a_i|) + \log \binom{|V|}{|a_i|} + \sum_{v \in a_i} L(v) \right], \tag{4}$$

where we first encode the number of subgroups $|S|$ using the universal code for integers, and then encode each subgroup description individually. For each description, first the number $|a_i|$ of variables used is encoded, then the set of variables using a uniform code over the set of all possible combinations of $|a_i|$ from $|V|$ variables, and finally the specific condition for a given variable. As we allow variables of three types, the latter is further specified by

$$L(v_{bin}) = \log 2 \,;\; L(v_{nom}) = \log |\mathcal{X}_v| \,;\; L(v_{num}) = \log N(n_{cut}), \tag{5}$$

where the code for each variable type assigns code lengths proportional to the number of possible partitions of the variable's domain. Note that this seems justified, as more partitions implies more potential spurious associations with the target that we would like to avoid. For `bin`ary variables only two conditions are possible, while for `nom`inal variables this is given by the size of the domain. For `num`eric variables it equals the number of possible combinations $N(n_{cut})$, as there can be conditions with one (e.g. $x \leq 2$) or two operators (e.g. $1 \leq x \leq 2$), which is a function of the number of possible subsets generated by n_{cut} cut points. Note that we here assume that equal frequency binning is used, which means that knowing X and n_{cut} is sufficient to determine the cut points.

3.3 Data Encoding

The remaining length function is that of the target data given the explanatory data and model, $L(Y \mid X, M)$. For this we first observe that for any given subgroup list of the form of Eq. (3), *any individual instance* $(\mathbf{x}_i, y_i)$ *is 'covered' by only one subgroup*. That is, the cover of a subgroup a_i, denoted D_i, depends on the order of the list and is given by the instances where its description occurs minus those instances covered by previous subgroups:

$$D_i = \{X_i, Y_i\} = \{(\mathbf{x}, y) \in D \mid a_i \sqsubseteq \mathbf{x} \wedge \left(\bigwedge_{\forall_{j<i}} a_j \not\sqsubseteq \mathbf{x} \right) \}. \tag{6}$$

Next, let $n_i = |D_i|$ be the number of instances covered by a subgroup (also known as *usage*). For a given subgroup a_i, we then estimate

$$\hat{\mu}_i = \frac{1}{n_i} \sum_{y \in Y_i} y \tag{7}$$

$$\hat{\sigma}_i^2 = \frac{1}{n_i} \sum_{y \in Y_i} (y - \hat{\mu}_i)^2, \tag{8}$$

where $\hat{\sigma}_i^2$ is the biased estimator such that the estimate times n_i equals the Residual Sum of Squares, i.e., $n_i\hat{\sigma}_i^2 = \sum_{y \in Y_i}(y - \hat{\mu}_i)^2 = RSS_a$.

Given the above, we can separately encode the covers of the individual subgroups, but we first show how to encode the target values not covered by any subgroup.

Encoding Target Values Not Covered by Any Subgroup. The target values not covered by any subgroup, given by $Y_d = \{(\mathbf{x}, y) \in D \mid \forall_{a_i \in M} a_i \not\sqsubseteq \mathbf{x}\}$, are covered by the default dataset 'rule' and distribution at the end of a subgroup list. As $\hat{f}_{d,\hat{\mu}_d,\hat{\sigma}_d}$ is known and constant for a given dataset, one can simply encode the instances using this (normal) distribution, resulting in encoded length

$$L(Y_d \mid \hat{\mu}_d, \hat{\sigma}_d) = \frac{n_d}{2}\log 2\pi + \frac{n_d}{2}\log\hat{\sigma}_d^2 + \left[\frac{1}{2\hat{\sigma}_d^2}\sum_{y \in Y_d}(y - \hat{\mu}_d)^2\right]\text{le}, \tag{9}$$

where $\text{le} = \log e$. The first two terms are normalizing terms of a normal distribution, while the last term represents the Residual Sum of Squares (RSS) normalized by the variance of the data. Note that when $Y_d = Y$, i.e., the whole dataset target, RSS is equal to $n_d\sigma_d$ and the last term reduces to $\text{le}n_d/2$.

Encoding Target Values Covered by a Subgroup. In contrast to the previous case, here *we do not know a priori the statistics defining the probability distribution corresponding to the subgroup*, i.e., $\hat{\mu}$ and $\hat{\sigma}$ are not given by the model and thus both need to be encoded. For this we resort to the Bayesian encoding of a normal distribution with mean μ and standard deviation σ unknown, which was shown to be asymptotically optimal [8]. An optimal code length is simply given by the negative logarithm of a probability, and the optimal Bayesian probability for Y_i is given by

$$P_{Bayes}(Y_i) = \int_{-\infty}^{+\infty}\int_{0}^{+\infty}(2\pi\sigma)^{-\frac{n_i}{2}}\exp -\frac{\sum_{y \in Y_i}(y-\mu)^2}{2\sigma^2}w(\mu,\sigma)\,\mathrm{d}\mu\,\mathrm{d}\sigma, \tag{10}$$

where $w(\mu, \sigma)$ is the prior on the parameters, which needs to be chosen.

The MDL principle requires the encoding to be as unbiased as possible for any values of the parameters, which leads to the use of uninformative priors. The most uninformative prior is Jeffrey's prior, which is $1/\sigma^2$ and therefore constant for any value of μ and σ, but unfortunately its integral is undefined, i.e., $\int\int \sigma^{-2}\,\mathrm{d}\sigma\,\mathrm{d}\mu = \infty$. Thus, we need to 1) constrain the parameter space and 2) make the integral finite, which we will do next in consecutive steps.

One of the best ways to constrain the parameter space without biasing it, is by multiplying Jeffrey's prior by a normal prior on the effect size, i.e., $\rho = \mu/\sigma \sim \mathcal{N}(0, \tau)$ [20]. We then still need to describe τ though; the most uninformative choice would be to use an inverse-chi-squared distribution, which would be equivalent to using a Cauchy prior on the effect size [20]. Unfortunately, this would lead to an open integral, which would render the approach infeasible for cases—like ours—where many probabilities need to be computed. The second best option is to fix $\tau = 1$, which gives a tractable formula that is equivalent to

introducing a virtual point and converges[4] to the Bayes Information Criterion (BIC) for large n. This is the best we can do and we proceed with this option.

Now, given the prior defined by $\rho = \mu/\sigma \sim \mathcal{N}(0,1)$, the remaining question is how we can make the integral over the prior finite. The most common solution, which we also employ, is to use k data points from Y_i, denoted Y_i^k, to create a proper conditional prior $w(\mu, \sigma \mid Y_i^k)$. As there are only two unknown parameters, we only need two points hence $k = 2$ [7,8]. Consequently, we first encode Y_i^2 with a non-optimal code that is readily available—here the encoding with the dataset distribution of Eq. (9)—and then use the Bayesian rule to derive the total encoded length of Y_i as

$$L(Y_i) = -\log \frac{P_{Bayes}(Y_i)}{P_{Bayes}(Y_i^2)} P(Y_i^2 \mid \mu_d, \sigma_d) = L_{Bayes}(Y_i) + L_{cost}(Y_i^2), \tag{11}$$

where $L_{cost}(Y_i^2) = L(Y_i^2 \mid \mu_d, \sigma_d) - L_{Bayes}(Y_i^2)$ is the extra cost incurred by encoding two points non-optimally. After some re-writing[5] we obtain the encoded length of the y values covered by a subgroup Y_i as

$$\begin{aligned} L(Y_i) &= L_{Bayes}(Y_i) + L_{cost}(Y_i^2) \\ &= 1 + \frac{n_i}{2}\log\pi - \log\Gamma\left(\frac{n_i}{2}\right) + \frac{1}{2}\log(n_i+1) + \frac{n_i}{2}\log n\hat{\sigma}_a^2 + L_{cost}(Y_i^2), \end{aligned} \tag{12}$$

where Γ is the Gamma function that extends the factorial to the real numbers ($\Gamma(n) = (n-1)!$ for integer n) and $\hat{\mu}_i$ and $\hat{\sigma}_i$ are the statistics of Equations (7) and (8), respectively. Note that for Y_i^2 any two unequal values (otherwise $\hat{\sigma}_2 = 0$ and $L_{Bayes}(Y_i^2) = \infty$) can be chosen from Y_i, thus we choose them such that they minimize $L_{cost}(Y_i^2)$. Finally, the total encoded size of Y is given by

$$L(Y \mid X, M) = \sum_{i \in M} L(Y_i) + L(Y_d \mid \mu_d, \sigma_d). \tag{13}$$

3.4 Properties and Quality Measure for Subgroup Lists

We next show[6] that the proposed data encoding is an instance of the classical definition of a quality measure as given by Eq. (1), and is tightly related to both an existing quality measure and the Bayesian two-sample t-test.

First, we show that Eq. (12)—with mean and variance unknown—converges, for large n, to Eq. (9)—with mean and variance known—plus an additional term. Using the Stirling approximation of $\Gamma(n+1) \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$ leads to

$$L(Y_i) \sim \frac{n_i}{2}\log 2\pi + \frac{n_i}{2}\log\hat{\sigma}_i^2 + \frac{n_i}{2}\mathrm{le} + \log\frac{n_i}{e}, \tag{14}$$

[4] See proof in Appendix 2 of the extended version [16].
[5] The full derivation of the Bayesian encoding and an in-depth explanation are given in Appendix 1 of the extended version [16].
[6] Derivations are given in Appendix 4 of the extended version [16].

where $\log \frac{n}{e}$ is equal to the penalty term of BIC and similar to the usual MDL complexity of a distribution [8].

Now, we can show that minimizing our MDL criterion is equivalent to maximizing a subgroup discovery quality function of the form Eq. (1). Focusing on the case where $S = \{s_1\}$ contains only one subgroup with statistics $\hat{\Theta}_1 = \{\hat{\mu}_1, \hat{\sigma}_1\}$, we start with $L(Y \mid X, M)$ (Eq. (2)), multiply it by minus one to make it a maximization problem, and add a constant $L(Y \mid \hat{\mu}_d, \hat{\sigma}_d)$, i.e., the encoded size of the whole target Y using the overall distribution dataset, to obtain

$$\begin{aligned} L(Y \mid \hat{\Theta}_d) - L(Y \mid X, M) &\sim n_i \left[\log \frac{\hat{\sigma}_d}{\hat{\sigma}_i} + \frac{\hat{\sigma}_i^2 + (\mu_1 - \mu_2)^2}{2\sigma_d^2} \mathrm{le} - \frac{\mathrm{le}}{2} \right] - \log(n_i) - L(S) \\ &= n_i D_{KL}(\hat{\Theta}_a; \hat{\Theta}_d) - \log(n_i) - L(S), \end{aligned} \tag{15}$$

where $\hat{\Theta}_a = \{\hat{\mu}_d, \hat{\sigma}_d\}$ and $n_i D_{KL}(\hat{\Theta}_a; \hat{\Theta}_d)$ is the usage-weighted Kullback-Leibler divergence between the normal distributions specified by the respective parameter vectors. This shows that *finding the MDL-optimal subgroup is equivalent to finding the subgroup that maximizes the weighted Kullback-Leibler (WKL) divergence*, an existing subgroup discovery quality measure [11] that was previously used for nominal targets, plus a term that defines the complexity of the subgroup. Moreover, note that Eq. (15) is equivalent to the Bayesian two-sample t-test [6] plus the complexity of the model, which plays the role of penalizing for multiple hypothesis testing. Finally, our measure is part of the family of *dispersion-corrected* subgroup quality measures, as it takes into account both the centrality and the spread of the target values [4].

Quality Measure for Subgroup Lists. Based on the previous, we naturally extend the KL-based measure for individual subgroups to subgroup lists and propose the Sum of Weighted Kullback-Leibler (SWKL) divergences:

$$\mathrm{SWKL}(S) = \sum_{a \in S} n_i D_{KL}(\hat{\Theta}_a; \hat{\Theta}_d) = \sum_{a_i \in S} n_i \left[\log \frac{\hat{\sigma}_d}{\hat{\sigma}_i} + \frac{\hat{\sigma}_i^2 + (\hat{\mu}_i - \hat{\mu}_d)^2}{2\hat{\sigma}_d^2} \mathrm{le} - \frac{\mathrm{le}}{2} \right] \tag{16}$$

An advantage of this measure is that it can not only be used for numeric targets, but for any type of probabilistic model. Note that computing SWKL is straightforward for subgroup lists as obtained by most methods, including ours, but not for subgroup sets as instances can be covered by multiple subgroups.

4 The SSD++ Algorithm

As the problem of finding an MDL-optimal list of subgroups is unfeasible, we propose a heuristic approach (as is common in MDL-based pattern mining [18,22]) based on Separate-and-Conquer (SaC) to construct the list, and beam-search to generate the subgroups to add at each iteration of SaC. The first reason for using greedy search to add one subgroup at the time, is its transparency, as it adds at

Algorithm 1: SSD++ algorithm

Data: Dataset D, number of cut points n_{cut}, beam width w_b, depth max. d_{max}
Result: Subgroup list S

1 $M \leftarrow [\Theta_d(Y)]$;
2 **repeat**
3 $\quad Cands \leftarrow BeamSearch(M, D, w_b, n_{cut}, d_{max})$;
4 $\quad s \leftarrow \arg\max_{\forall s' \in Cands} : \delta L(D, M \oplus s')$;
5 $\quad M \leftarrow M \oplus s$;
6 **until** $\delta L(D, M \oplus s') \leq 0, \forall s' \in Cands$;

each iteration the locally best subgroup found by the beam search. Beam-search, on the other hand, was empirically shown, in the context of subgroup discovery for numeric targets, to be very competitive in terms of quality when compared to a complete search with an associated speedup improvement [14]. Also, its straightforward implementation allows to easily extend this framework to other types of targets, not just numeric. To quantify the quality of annexing $\oplus$ a subgroup s at the end (after all the other subgroups) of model M, we use the *normalized gain* $\delta L(M \oplus s) = (L(D, M) - L(D, M \oplus s))/n_s$, which was first introduced in the classification setting and proved to outperform its non-normalized version in that setting [18]. For a detailed empirical comparison of normalized gain and its non-normalized version please refer to Appendix 6 [16].

Algorithm 1 presents SSD++, a greedy algorithm that iteratively adds subgroups to an empty subgroup list until no more compression can be gained, where compression is measured in terms of normalized gain of adding a subgroup s.

The *beam search algorithm* starts by discretizing all variables depending on their subsets, i.e. categorical and binary with the operator *equal to* (=) and numeric by generating all subsets with n_{cut} points. At each iteration the w_b subgroups that maximize the selected gain are chosen and will be expanded with all discretized variables until the maximum depth d_{max} of the description is achieved.

The *SSD++ algorithm* [15] takes as input the dataset D, and the beam search parameters, namely the number of cut points n_{cut}, the width of the beam w_b, and the maximum depth of search d_{max}. The algorithm starts by adding the dataset empirical distribution to the model (Ln 1). Then, while there is a subgroup that improves compression (Ln 6), it keeps iterating over three steps: 1) generating the candidates using beam search (Ln 3); 2) finding the subgroup that maximizes the normalized gain (Ln 4); and 3) adding that subgroup to the end of the model, i.e., after all the existing subgroups in the model (Ln 5). The beam search returns the best subgroup according to the data not covered by any subgroup in the model M and its parameters (w_b, n_{cut}, d_{max}). When there is no subgroup that improves compression (non-positive gain) the while loop stops and the subgroup list is returned. Note that beam search is used at each iteration, instead of only once at the beginning, as it can converge to local optima, and would thus bias our search to the top-k subgroups instead of the best at each iteration.

5 Experiments

We evaluate SSD++[7] by comparing it to 1) a classical top-k mining algorithm, as a baseline of a non-diverse method, and 2) the sequential covering algorithm, henceforth called top-k and seq-cover respectively, which are both available in the implementation of the DSSD algorithm[8].

DSSD and SISD will not be compared due to two interconnected issues: 1) the lack of a *global* definition of the optimal set for a dataset; 2) the absence of a definition for the interaction between subgroups that overlap. The first issue has as a natural consequence that none of the methods have a clear stopping criteria as the definition of when a set describes the data well is not available, apart from the user-specified hyperparameter 'number of subgroups'. Added to this, both issues give rise to the question of how to measure the interaction of subgroups in the region of their overlap from a model (global) perspective, i.e., they could behave as an additive or a multiplicative mixture of their probabilities for example. These issues hamper the comparison with both methods as they do not have a clear stopping criteria and a formulation of their overlap interaction, of which the latter is necessary for our proposed measure SWKL. On the other hand, a direct use of SWKL assuming a list formulation, i.e. ordering them and removing the overlap, will always rate them lower, which was corroborated with our initial experiments. Moreover, we do not compare with prediction algorithms that generate rules for regression, such as RIPPER or CART, as the rules generated aim at making the best prediction possible, and not the highest difference from the dataset distribution, as shown theoretically in Appendix 5 [16].

Data. We use a set of 16 benchmark datasets from the Keel[9] repository commonly used for subgroup discovery. The complete description of the datasets is given in Table 2; the datasets were chosen to be diverse, ranging from 297 to 22 784 instances and from 2 to 40 variables.

Hyperparameter Selection. *SSD++:* the algorithm admits as hyperparameters: the width of the beam w_b; number of cut points n_{cut}; and maximum depth of search d_{max}. By varying these parameters over the datasets the results can be seen in Appendix 7 [16] and it was concluded that: 1) no descriptions of size much greater than 5 are found; 2) after $n_{cut} = 5$ (the default value for seq-cover) the subgroups returned are virtually the same but with numerical values refined; 3) for most datasets the quality of the subgroup list stabilizes beyond $w_b = 100$. Thus, for the rest of our experiments the parameters are set accordingly.

Top-k: the software used here is the top-k subgroups implemented in DSSD, which is equivalent to most top-k subgroup miners. As it is common with top-k miners a depth-first search is used for small datasets $|D| \leq 2000$ and a beam

[7] For the implementation of SSD++ and to reproduce the experiments see Proença [15].
[8] http://www.patternsthatmatter.org/software.php#dssd/.
[9] http://www.keel.es/.

Table 2. Dataset properties: number of instances, and variables.

Dataset	$\|D\|$	categorical	numerical	Dataset	$\|D\|$	categorical	numerical
cholesterol	297	7	5	wizmir	1 461	0	9
baseball	337	4	12	abalone	4 177	0	8
autoMPG8	392	0	6	puma32h	8 192	0	32
dee	365	0	6	ailerons	13 750	0	40
ele-1	495	0	2	elevators	16 599	0	18
forestFires	517	0	12	bikesharing	17 379	2	10
concrete	1 030	0	8	california	20 640	0	8
treasury	1 049	0	15	house	22 784	0	16

search for the rest. For the quality measure it uses the Weighted Kullback-Leibler without dispersion, i.e., $WKL_\mu(s) = n_s/\hat{\sigma}_d(\hat{\mu}_d - \hat{\mu}_s)^2$ as described in Appendix 3 [16], as the algorithm does not accept its dispersion-aware version used in Eq. (16). Also, as it does not have a termination criteria, the k number of subgroups returned is selected as the number of subgroups found by SSD++.

Seq-cover: to ensure fairness the same beam search hyperarameters as SSD++ are used, i.e., $d_{max} = 5$, $w_b = 100$, $n_{cut} = 5$. As quality measure it uses the Weighted Kullback-Leibler without dispersion for the same reasons as top-k.

5.1 Subgroup List Quality

The results can be seen in Table 3, and Figs. 1 and 2. The algorithms are compared in terms of Sum of Weighted Kullback-Leibler (SWKL) of Eq. (16) for the quality of the list, number of subgroups $|S|$, average number of variables per description $|a|$, standard deviation of the first subgroup $\tilde{\sigma}_{top1}$, runtime and average Jaccard index of the lists. Note that $\tilde{\sigma}_{top1}$ shows the most important characteristic first found by each miner. In the case of the averaged Jaccard index it is computed based on the average of the Jaccard index between the 1-vs-1 covers (when considered independently) of the subgroups in the list, i.e., for the case of a list of 4 subgroups, 6 values are averaged.

From Table 3 we see that SSD++ obtains the best score in terms of our proposed measure SWKL for 12 out of 16 datasets. As expected the top-k algorithm obtains a lower score for all datasets except for one. This supports that our proposed measure SWKL gives weight to subgroup sets that cover different parts of the dataset. Also, in terms of the dispersion of the first subgroup its value is lower for 80% of the cases. In terms of the number of rules and compared with seq-cover, SSD++ tends to find fewer subgroups for smaller datasets ($|D| \leq 10\,000$), and more for larger datasets. For the latter, the experiments showed that on average each subgroup covers more than 100 instances per subgroup. In terms of the number of variables per description, it tends to find more compact descriptions than top-k and seq-cover.

In terms of runtime, as per Fig. 1, SSD++ has a similar performance to seq-cover for small sample sizes ($|D| \leq 1000$) and 10 times slower for larger sizes.

Table 3. Performance results of {Summed Weighted Kullback-Leibler Divergence (SWKL) divided by number of examples; standard deviation of the first subgroup normalized by σ_d; number of subgroups; average number of conditions per subgroup description} per dataset for each algorithm.

	top-k				seq-cover				SSD++			
datasets	SWKL	$\tilde{\sigma}_{top1}$	$\|S\|$	$\|a\|$	SWKL	$\tilde{\sigma}_{top1}$	$\|S\|$	$\|a\|$	SWKL	$\tilde{\sigma}_{top1}$	$\|S\|$	$\|a\|$
cholesterol	0.14	1.49	1	5	**0.84**	**1.51**	33	4	0.11	1.99	1	3
baseball	0.25	0.85	8	5	1.69	0.82	26	4	**1.92**	**0.22**	8	2
autoMPG8	0.48	0.54	10	5	1.36	0.54	22	3	**1.65**	**0.18**	10	2
dee	0.49	0.47	8	5	**1.47**	0.50	20	4	1.33	**0.44**	8	2
ele-1	0.29	**1.06**	9	3	1.14	**1.06**	22	3	**1.25**	1.33	9	2
forestFires	0.58	6.84	23	5	2.85	6.84	57	4	**3.80**	**0.03**	23	3
concrete	0.25	0.78	19	5	**1.27**	0.65	35	4	**1.27**	**0.34**	19	3
treasury	0.42	0.70	31	5	2.41	0.68	25	3	**3.73**	**0.05**	31	2
wizmir	0.77	0.31	22	5	2.17	0.31	26	4	**2.73**	**0.16**	22	2
abalone	0.23	0.59	25	5	0.48	0.59	118	3	**0.71**	**0.45**	25	3
puma32h	0.55	0.59	42	5	**1.48**	0.59	76	5	1.42	**0.30**	42	3
ailerons	0.24	1.23	19	2	1.04	1.23	101	4	**1.58**	**1.10**	197	4
elevators	0.25	**1.44**	141	4	0.84	**1.44**	157	4	**1.30**	**1.44**	160	4
bikesharing	0.27	1.09	127	5	1.24	1.09	91	4	**1.68**	**0.07**	127	4
california	0.19	0.90	163	4	0.70	0.90	135	5	**1.15**	**0.84**	163	4
house	0.19	**1.59**	280	5	0.91	**1.59**	145	4	**2.08**	2.18	280	5

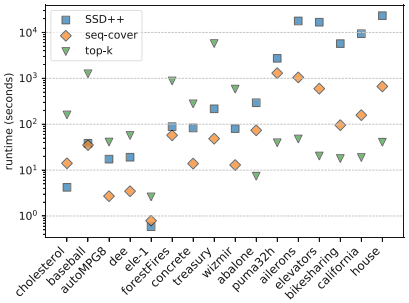

Fig. 1. Runtime in seconds per algorithm and dataset.

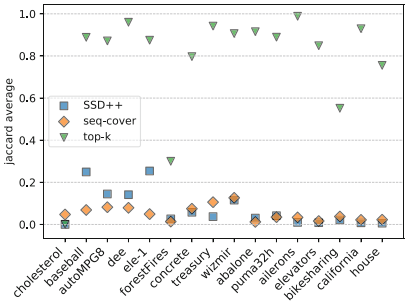

Fig. 2. Average overlap of subgroups in a list per dataset and algorithm.

This can, in part, be explained, by the larger number of subgroups found for these datasets—from 1.2 to 4 times more. Figure 2 shows that for small datasets the overlap is larger than for seq-cover, while for larger datasets our formulation tends to have similar level of overlap.

6 Case Study: Hotel Bookings

To test the usefulness of our method we applied it to the problem of understanding the type of clients that make a hotel booking based on how much time in advance (lead time in days) this was done. To this end we used the "Hotel booking demand dataset" [1], and analysed the data referent to a *resort hotel* in the year of 2016. The first four subgroups of a total of 260 obtained with SSD++ can be seen in Fig. 1 (in Sect. 1) and its subgroups versus the dataset in Fig. 3. Only the first 4 subgroups are shown here for clarity, and given that greedy search is used, they are also the 4 most interesting subgroups.

The results show us a detailed picture of the dataset and at first glance, one notices that most subgroups cover a small number of instances. Nevertheless, this is normal as they represent highly defined subgroups, with a very different mean and an almost zero standard deviation, compared with the dataset $\hat{\mu}_d = 92$ and $\hat{\sigma}_d = 99$. As an example, subgroup 1 has an average lead time circa 6 times higher than the dataset distribution, together with a standard deviation that is 3 times smaller. This subgroup seems to represent a group of people that travelled together from Great Britain and all chose the same type of booking, while with some slight days of difference in their bookings. Another interesting subgroup is the 4^{th} which shows that there is a group of around 20 similar bookings for groups of 2 or more adults done with only 9 days before arrival when the deposit type is refundable. If one would follow the whole subgroup list one would have a complete summary of the bookings done.

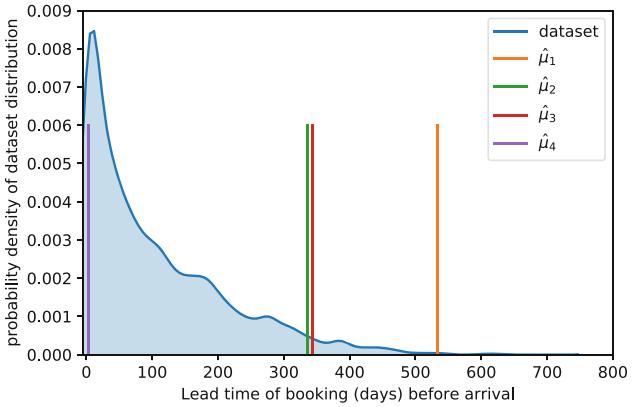

Fig. 3. Kernel density estimation of the dataset distribution and mean value of the first 4 found subgroups.

7 Conclusions

We introduced a dispersion-aware problem formulation for subgroup set discovery based on subgroup lists, the MDL principle, and Bayesian statistics. We

proved our formulation to be equivalent to an existing subgroup quality measure for the case of finding the single best subgroup, and showed a relationship to Bayesian testing. Based on these insights we proposed a new evaluation measure for subgroup lists, the sum of Weighted Kullback-Leibler divergences (SWKL).

To find good subgroup lists we introduced SSD++, a greedy algorithm that we empirically evaluated on 16 datasets and compared against state-of-the-art algorithms. SSD++ was shown to outperform the other methods in terms of both our proposed measure and subgroup set complexity as quantified by subgroup and/or description sizes, and discovers subgroups with small standard deviation.

Acknowledgment. This work is part of the research programme Indo-Dutch Joint Research Programme for ICT 2014 with project number 629.002.201, SAPPAO, which is financed by the Netherlands Organisation for Scientific Research.

References

1. Antonio, N., de Almeida, A., Nunes, L.: Hotel booking demand datasets. Data Brief **22**, 41–49 (2019)
2. Atzmueller, M.: Subgroup discovery. Wiley Interdisc. Rev. Data Min. Knowl. Disc. **5**(1), 35–49 (2015)
3. Belfodil, A., et al.: FSSD-a fast and efficient algorithm for subgroup set discovery. In: Proceedings of DSAA 2019 (2019)
4. Boley, M., Goldsmith, B.R., Ghiringhelli, L.M., Vreeken, J.: Identifying consistent statements about numerical data with dispersion-corrected subgroup discovery. Data Min. Knowl. Disc. **31**(5), 1391–1418 (2017). https://doi.org/10.1007/s10618-017-0520-3
5. Bosc, G., Boulicaut, J.F., Raïssi, C., Kaytoue, M.: Anytime discovery of a diverse set of patterns with Monte Carlo tree search. Data Min. Knowl. Disc. **32**(3), 604–650 (2018). https://doi.org/10.1007/s10618-017-0547-5
6. Gönen, M., Johnson, W.O., Lu, Y., Westfall, P.H.: The Bayesian two-sample t test. Am. Stat. **59**(3), 252–257 (2005)
7. Grünwald, P., Roos, T.: Minimum description length revisited. Int. J. Math. Ind. **11**(1), 1930001 (29 p.) (2019)
8. Grünwald, P.D.: The Minimum Description Length Principle. MIT Press, Cambridge (2007)
9. Klösgen, W.: Explora: a multipattern and multistrategy discovery assistant. In: Advances in Knowledge Discovery and Data Mining, pp. 249–271 (1996)
10. Lavrač, N., Kavšek, B., Flach, P., Todorovski, L.: Subgroup discovery with CN2-SD. J. Mach. Learn. Res. **5**, 153–188 (2004)
11. van Leeuwen, M.: Maximal exceptions with minimal descriptions. Data Min. Knowl. Disc. **21**(2), 259–276 (2010). https://doi.org/10.1007/s10618-010-0187-5
12. van Leeuwen, M., Knobbe, A.: Diverse subgroup set discovery. Data Min. Knowl. Disc. **25**(2), 208–242 (2012). https://doi.org/10.1007/s10618-012-0273-y
13. Lijffijt, J., Kang, B., Duivesteijn, W., Puolamaki, K., Oikarinen, E., De Bie, T.: Subjectively interesting subgroup discovery on real-valued targets. In: 2018 IEEE ICDE, pp. 1352–1355. IEEE (2018)
14. Meeng, M., Knobbe, A.: For real: a thorough look at numeric attributes in subgroup discovery. Data Min. Knowl. Disc. **35**(1), 158–212 (2021)

15. Proença, H.M. : HMProenca/SSDpp-numeric: v2020.06.0 (2020). https://github.com/HMProenca/SSDpp-numeric. Archived at https://doi.org/10.5281/zenodo.3901236
16. Proença, H.M., Grünwald, P., Bäck, T., van Leeuwen, M.: Discovering outstanding subgroup lists for numeric targets using MDL. Preprint arXiv:2006.09186 (2020)
17. Proença, H.M., Klijn, R., Bäck, T., van Leeuwen, M.: Identifying flight delay patterns using diverse subgroup discovery. In: 2018 SSCI, pp. 60–67. IEEE (2018)
18. Proença, H.M., van Leeuwen, M.: Interpretable multiclass classification by MDL-based rule lists. Inf. Sci. **512**, 1372–1393 (2020)
19. Rissanen, J.: Modeling by shortest data description. Automatica **14**(5), 465–471 (1978)
20. Rouder, J.N., Speckman, P.L., Sun, D., Morey, R.D., Iverson, G.: Bayesian t tests for accepting and rejecting the null hypothesis. Psychon. Bull. Rev. **16**(2), 225–237 (2009)
21. Van Leeuwen, M., Galbrun, E.: Association discovery in two-view data. IEEE Trans. Knowl. Data Eng. **27**(12), 3190–3202 (2015)
22. Vreeken, J., Van Leeuwen, M., Siebes, A.: KRIMP: mining itemsets that compress. Data Min. Knowl. Disc. **23**(1), 169–214 (2011). https://doi.org/10.1007/s10618-010-0202-x